

Introduction to Pattern Recognition

**Forensics investigation from fingerprint microbes**

**Yasin Samet Karaca**

December 2022

I wanted to show the analysis of the best percentage result using the given data set using the Python program in the project. In order to achieve the best success result, it is aimed to reach the best percentage result by trying different algorithms and different parameters within these algorithms. Although some algorithms work fast, they give low percentage results, while some slower algorithms give high percentage results. With this project it is expected to have the highest possible correct classification percentage. In order to achieve that you are expected to perform attribute selection (note: cross-validation in attribute selection is also required), and then go for classification with the selected attributes.

The algorithms I use are as follows.

- Ada Boost Classifier

- Logistic Regression

- Gradient Boosting Algorithm

- K-Nearest Neighbours

- Random Forest Classifier

- Xgboost

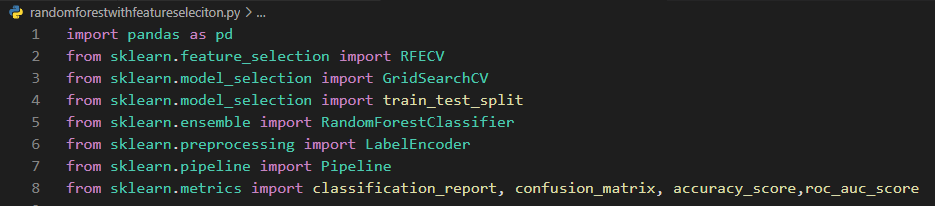
- Naive Bayes

- Decision Tree

**Random Forest Classifier**

Firstly, I use Random Forest Classifier Algorithm.

**PluginS that we need to uSe**

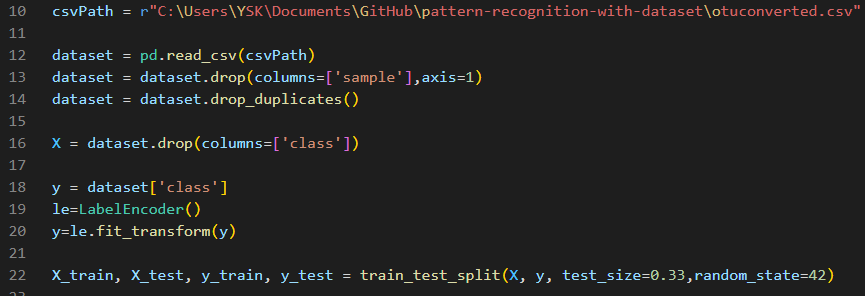
****

Here, need to set a path to read csv file.

Random\_state: Controls the shuffling applied to the data before applying the split.

Test size = 0.33

Left and right results are converted to numbers 0 and 1 with the Label encoder.



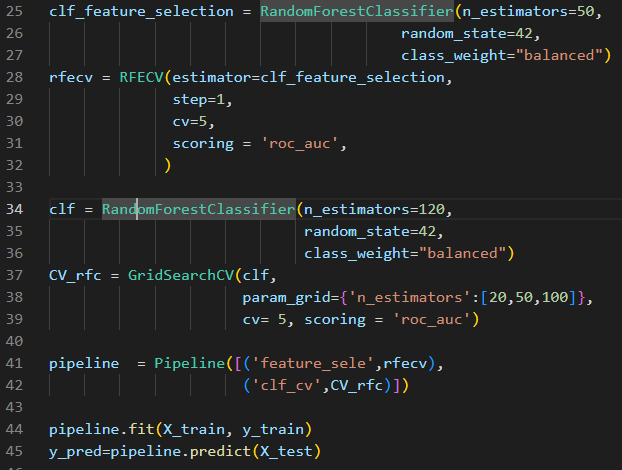
RFECV (Recursive Feature Elimination with Cross-Validation) performs recursive feature elimination with cross-validation loop to extract the optimal features. Scikit-learn provides RFECV class to implement RFECV method to find the most important features in a given dataset.

GridSearchCV is the process of performing hyperparameter tuning in order to determine the optimal values for a given model. As mentioned above, the performance of a model significantly depends on the value of hyperparameters. Note that there is no way to know in advance the best values for hyperparameters so ideally, we need to try all possible values to know the optimal values. Doing this manually could take a considerable amount of time and resources and thus we use GridSearchCV to automate the tuning of hyperparameters.

Basically, I want to fine tune the hyper parameter of our classifier (with Cross validation) after feature selection using recursive feature elimination (with Cross validation).

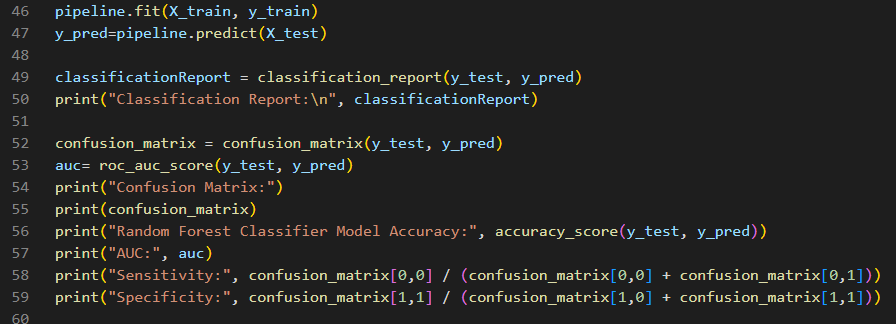
Pipeline object is exactly meant for this purpose of assembling the data transformation and applying estimator. It would be possible with the following approach

The purpose of the pipeline is to assemble several steps that can be cross-validated together while setting different parameters.



Cross validation in feature selection = 5

So right now I have test and predicdion data. I use them to create Confusion Matrix for classificafion report and accuracy.



I have everything we need to find Sensitivity, auc and Specificity. Now take a look at the formula.

**Sensitivity = (True Positive)/(True Positive + False Negative)**

**Specificity = (True Negative)/(True Negative + False Positive)**

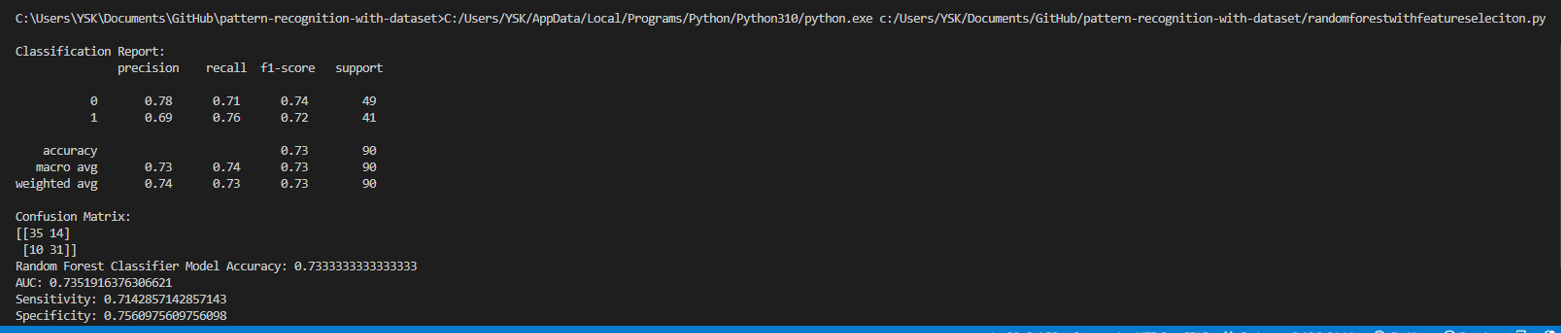
To find out how can we find true positive, true negative, false positive and false negative we can use this table.



**Sensitivity = cMatrix[0, 0]/(cMatrix[0, 0]+cMatrix[0, 1])**

**Specificity = cMatrix[1, 1]/(cMatrix[1, 0]+cMatrix[1, 1])**

**Converted to numbers 0 and 1 to left and right hand results with Label encoder**

**ClaSSification Report: **

**Accuracy: 0,733**

**Auc: 0,7351**

**Sensitivity: 0,714**

**Specificity: 0,7556**

**Random Forest Classifier with My Cross Validation in Feature Selection**

I used Stratified K-Fold cross-validation to evaluate the attribute selection

**Steps**

Importing Libraries and reading dataset.

Data preprocessing. Dividing the Data and Encoding the Labels.

Create a random forest classifier

Use Stratified K-Fold cross-validation to evaluate the attribute selection

Fit the classifier to the training data and get the feature importances

Select the top N most important features

Select the top N most important features from the entire dataset

Fit the classifier to the entire dataset using the selected features

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Cross validation in feature selection = 5

**Random Forest Classifier without Feature Selection**

Random Forest is a supervised machine learning algorithm which is based on ensemble learning.

**Steps**

Importing Libraries and reading dataset.

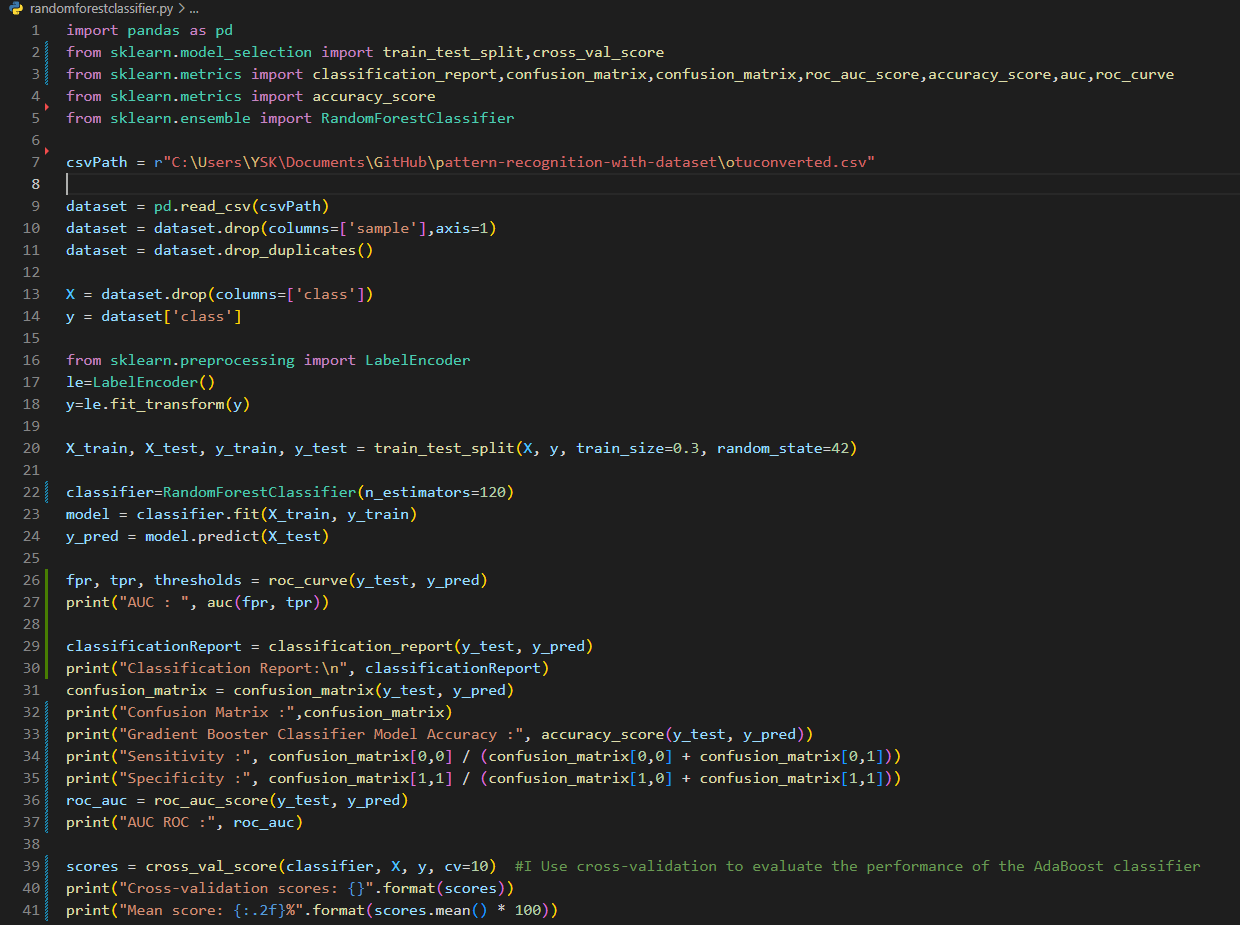
Data preprocessing. Dividing the Data and Encoding the Labels.

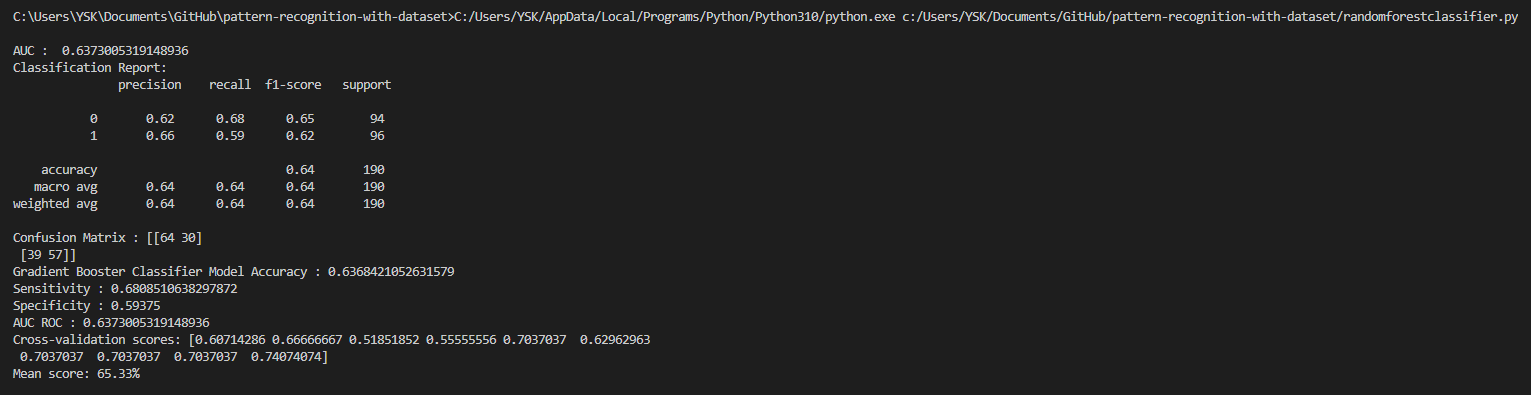
Splitting dataset into Training and Testing Set.

Implementing a Random forest classifier.

Predicting test cases using random forest.

Checking the scores.



****

Cross validation in feature selection = 5

Accuracy: 0,63824

Sensitivity: 0,8

Specificity: 0,59

Cross Validation Mean Score:0,6533

Auc: 0,637

**K Nearest Neighbours without Feature Selection**

KNN classifier is one of the most sophisticated and widely used classification algorithm. Some of the features that make this algorithm so popular are as mentioned below:

KNN does not make any underlying assumption about the data.

It gives the user the flexibility to choose the distance measure metric.

It has a relatively higher accuracy than many classification algorithms.

With the addition of more data points, the classifier constantly evolves and is capable of quickly adapting to the changes in input dataset.

**Standart Scaler** : Standardize features by removing the mean and scaling to unit variance.

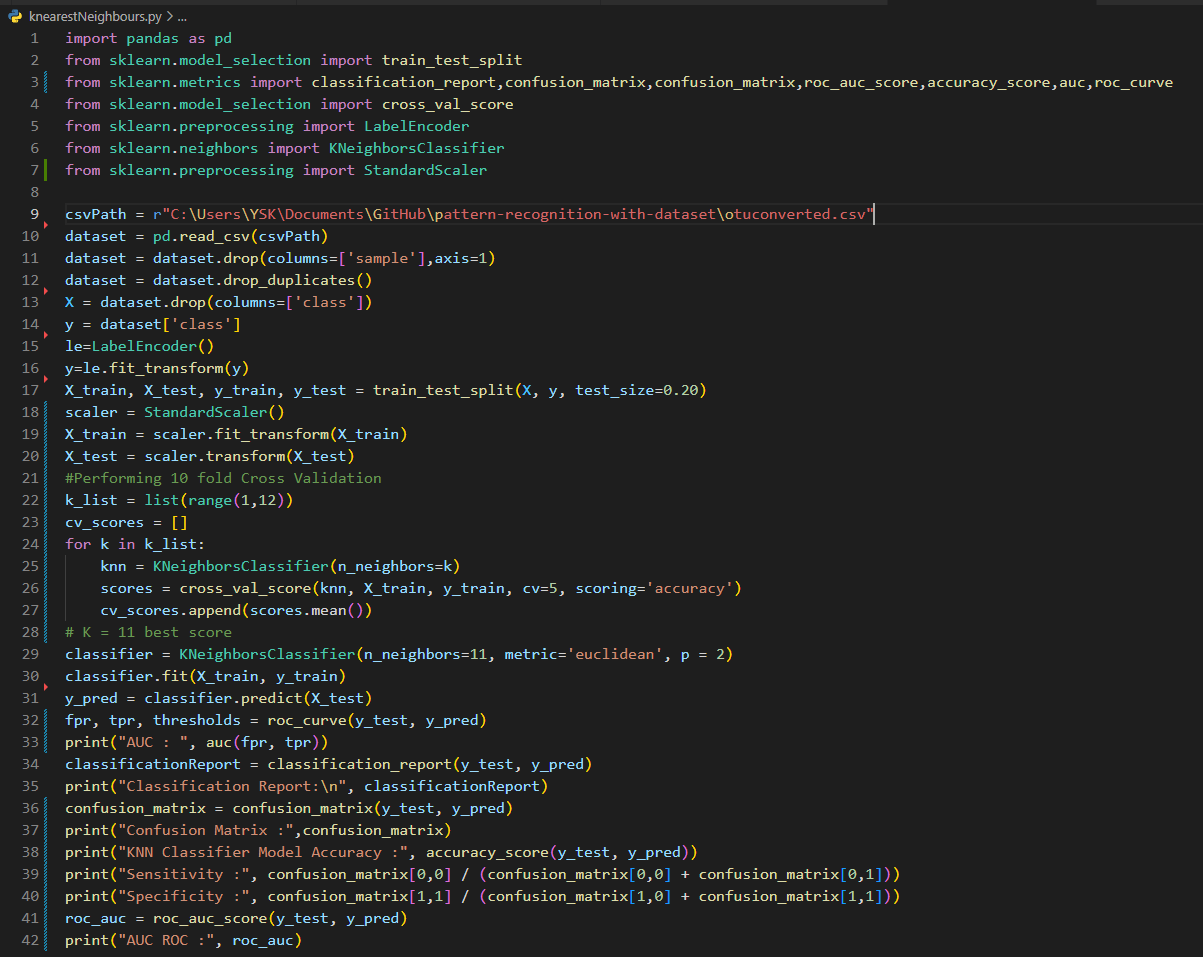
**Steps**

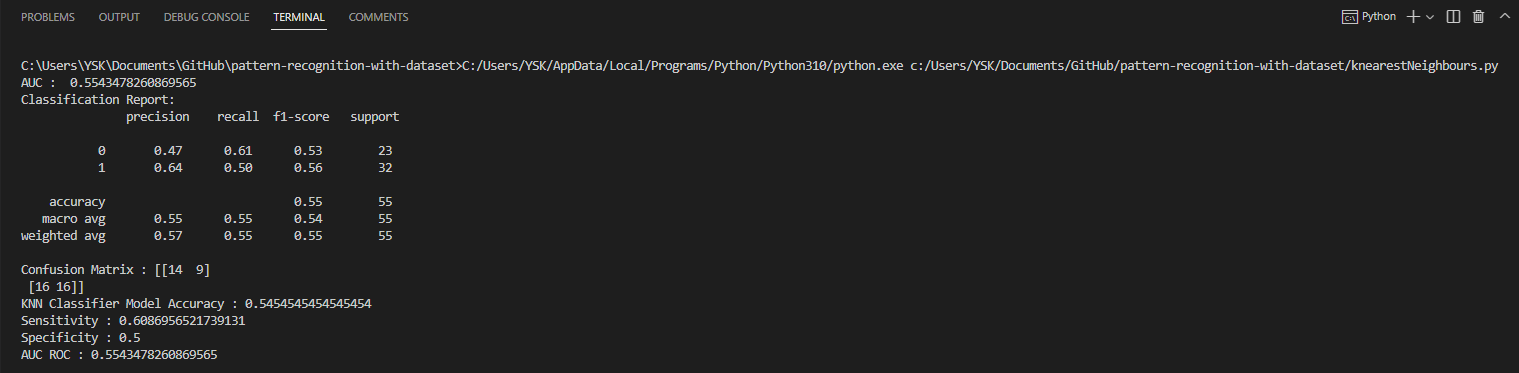
Importing the Required Libraries and Loading the Dataset

Dividing the Data and Encoding the Labels (Label Encoding)

Splitting the Dataset, Feature Scaling and Fitting the Model

Evaluating the Predictions and Cross Validating



****

**Accuracy: 0,54**

**Auc: 0,554347**

**Sensitivity: 0,605**

**Specificity: 0,5**

**K Nearest Neighbours With Feature Selection**

Sequential Feature Selector adds (forward selection) or removes (backward selection) features to form a feature subset in a greedy fashion. At each stage, this estimator chooses the best feature to add or remove based on the cross-validation score of an estimator.

We have added our features to another csv file so that our algorithm can make feature selection.

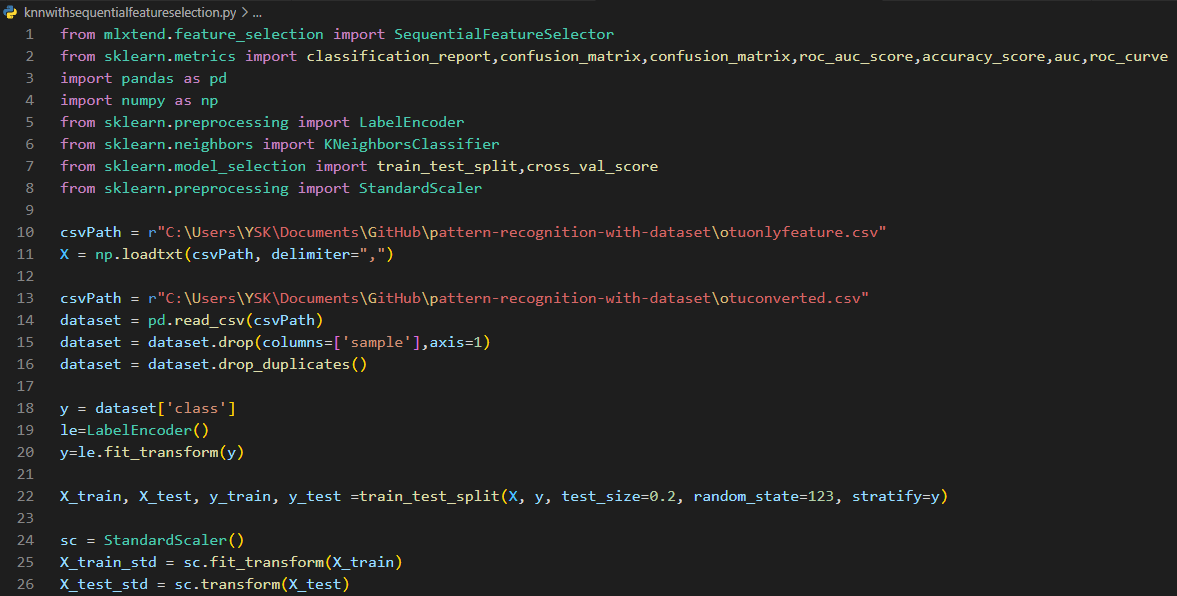
**Steps**

Import dataset and libraries

Drop Duplicate columns.

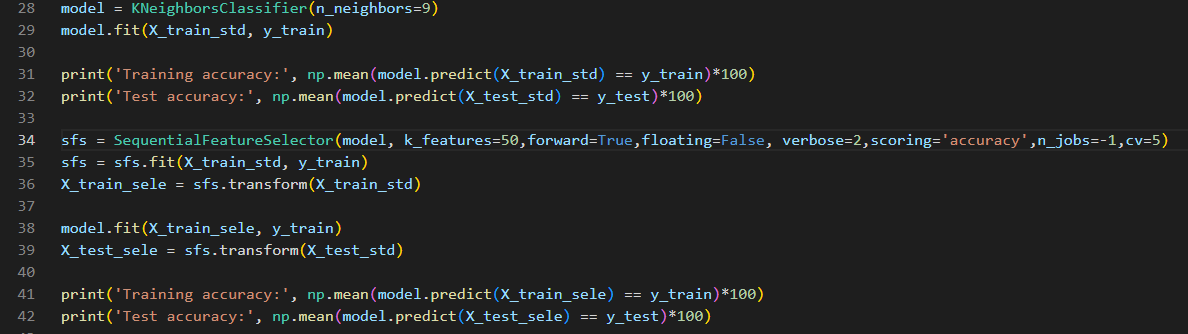
Split data into separate training and test set

Feature Scaling and engineering



Feature Selection.

Model training



**Training accuracy: 61.57407407407407**

**Test accuracy: 60.0**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

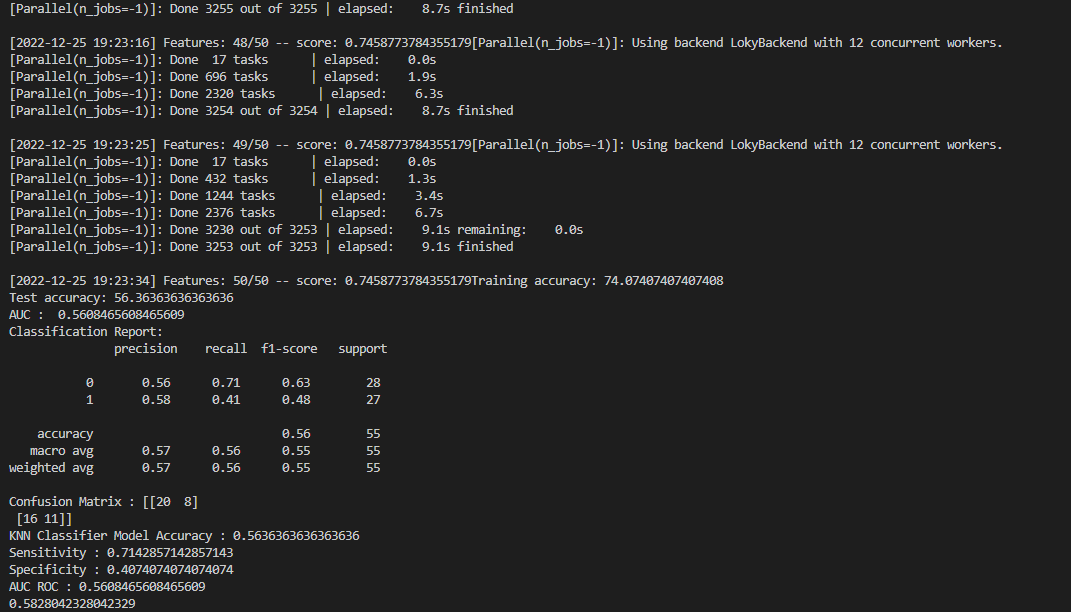
When k\_features = 50.

**Accuracy: 0,5936**

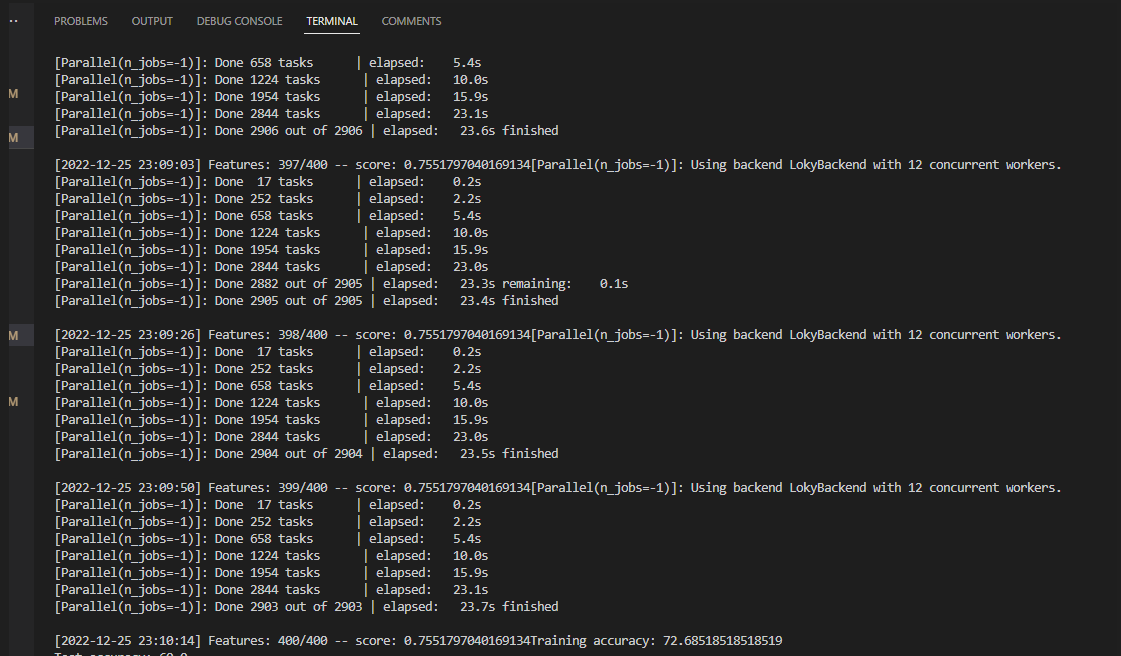
**Auc: 0,56**

**Sensitivity: 0,714**

**Specificity: 0,40**

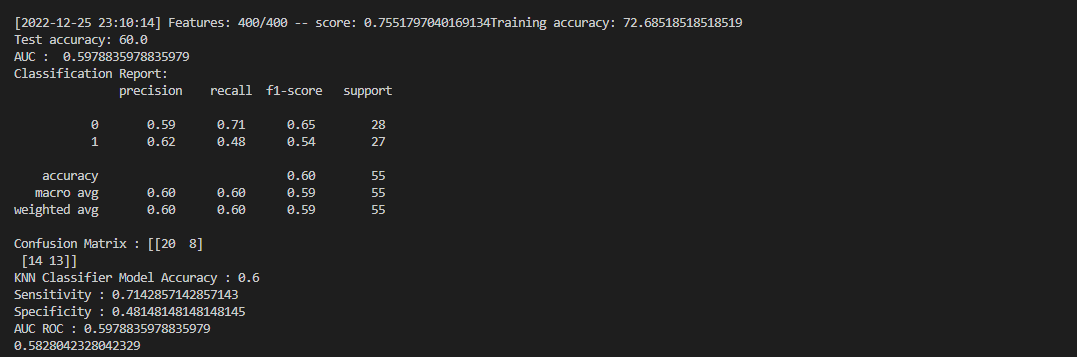
****

50 features are selected

****

When k\_features = 400.

400 features are selected

****

When k\_features = 50.

Accuracy: 0,6

Auc: 0,58

Sensitivity: 0,714

Specificity: 0,48

Trainin Accuracy : %72,68

Test Accuracy : %60.0

**Ada Boost Classifier Without Feature Selection**

It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier.

It works in the following steps:

Initially, Adaboost selects a training subset randomly.

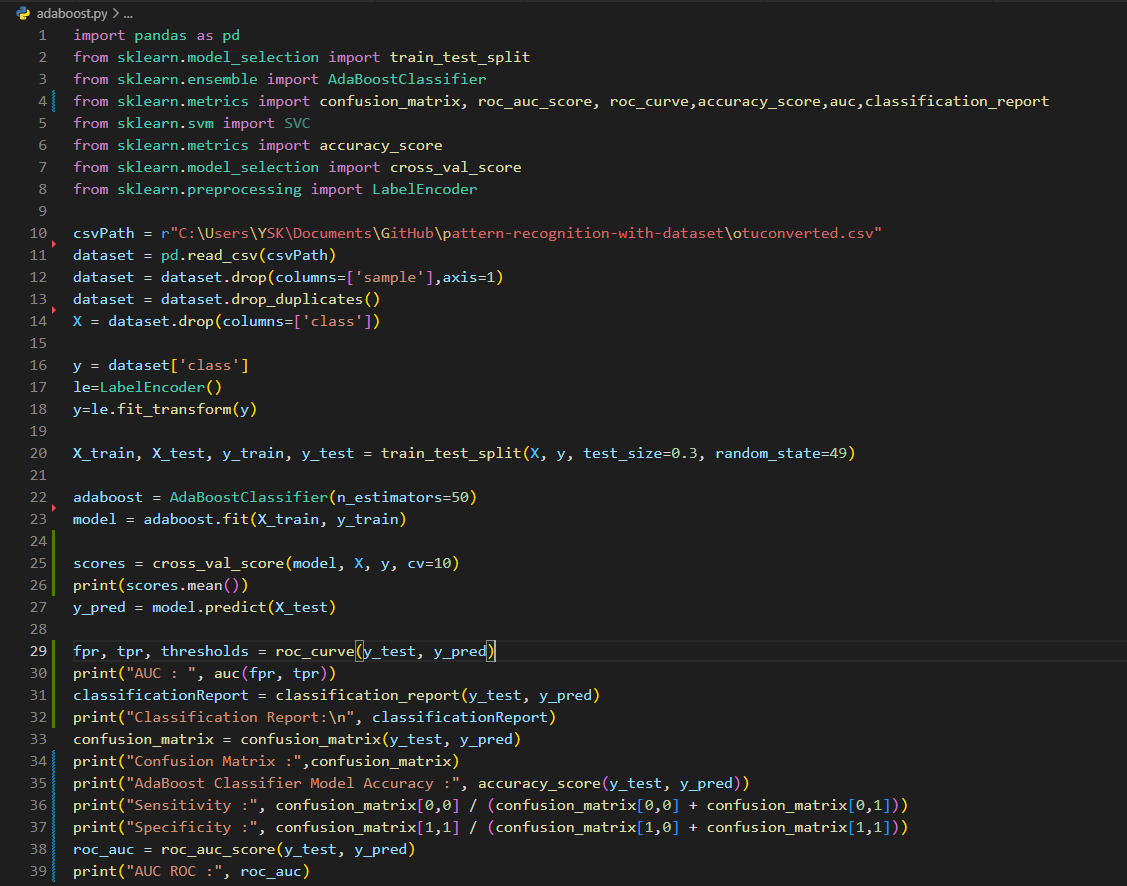
It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.

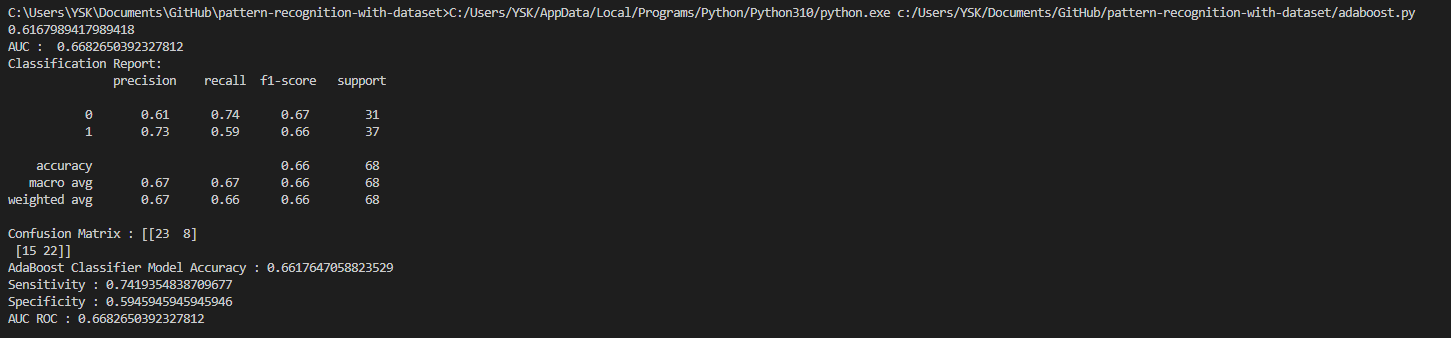
It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.

Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.

This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.

To classify, perform a "vote" across all of the learning algorithms you built.





Accuracy : 0,661

Sensitivity : 0,74

Specificity :0,59

Auc: 0,66896

**Ada Boost Classifier With Feature Selection**

Recursive feature elimination with cross-validation to select features.

Gridsearchcv algorithm is used for parameter selection with cross validation and rfecv algorithm is used for feature selection.

**Steps**

Import dataset and libraries.

Drop Duplicate columns and use label encoder.

Split the data into a training set and a test set.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Create the grid search object.

Fit the grid search object to the training data and use the best parameters to create the final AdaBoost classifier .metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

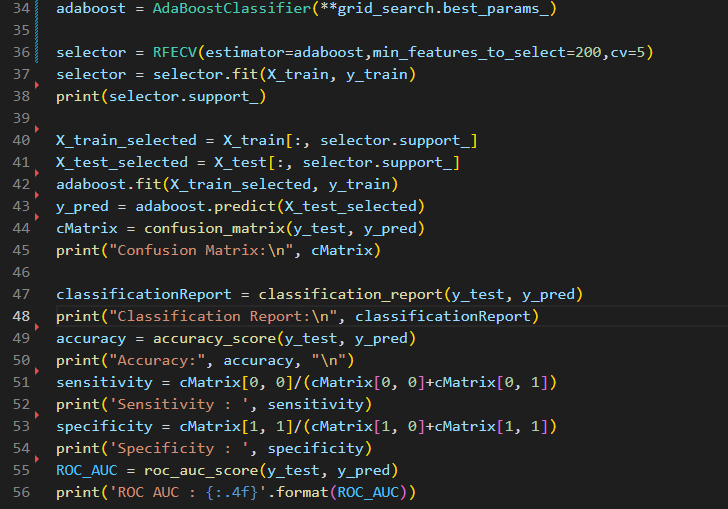
Best parameters: {'learning\_rate': 0.1, 'n\_estimators': 200}

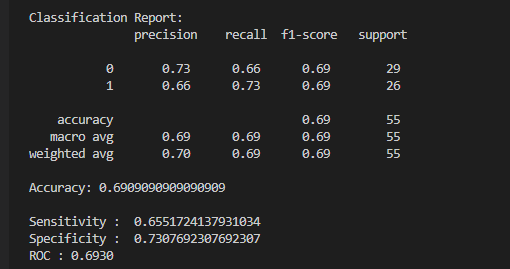
Use RFECV to select the best attributes. Print the selected attributes

Get the selected features. Train the AdaBoost classifier on the selected features

Make predictions on the test set

Calculate the accuracy of the predictions





Accuracy : 0,69090900909090

Sensitivity : 0,65

Specificity :0,73

Auc: 0,6930

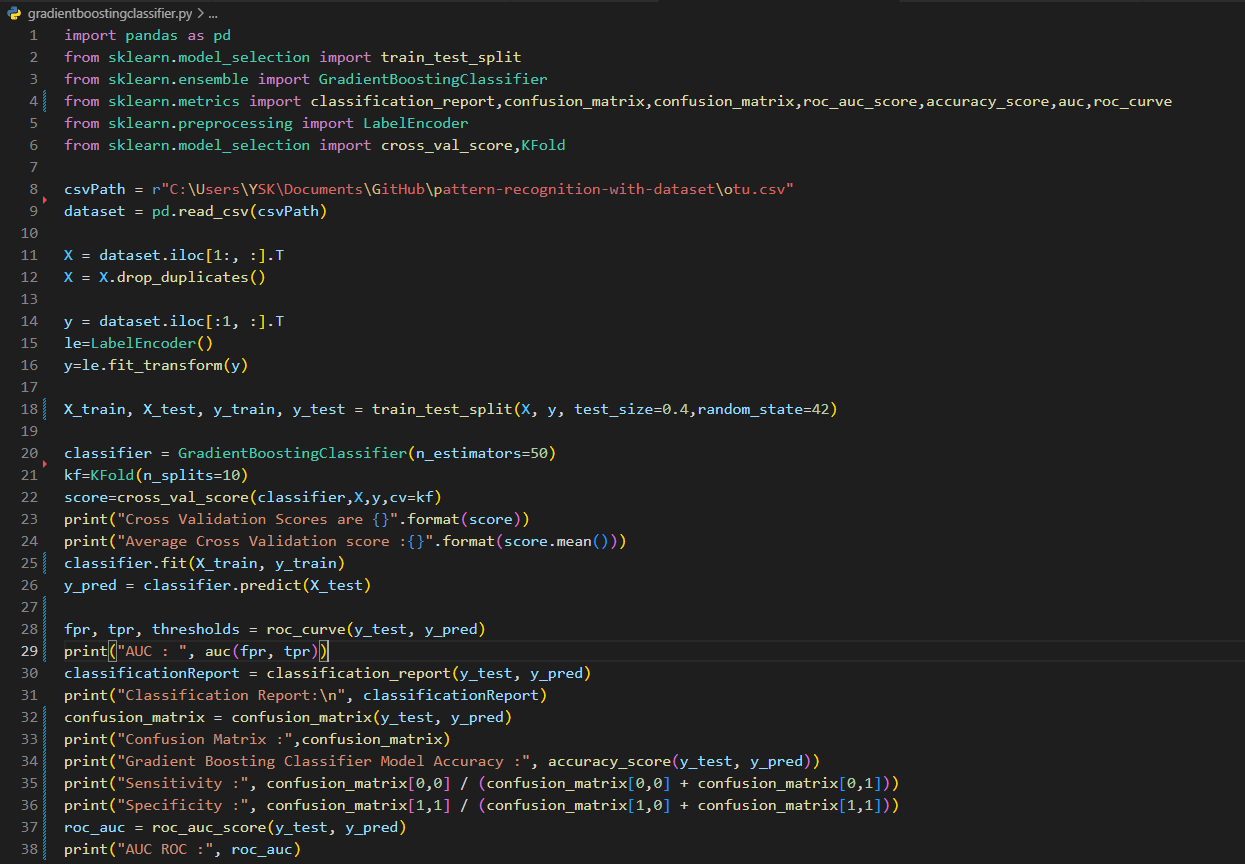
**Gradient Booster Algorithm Without Feature Selection**

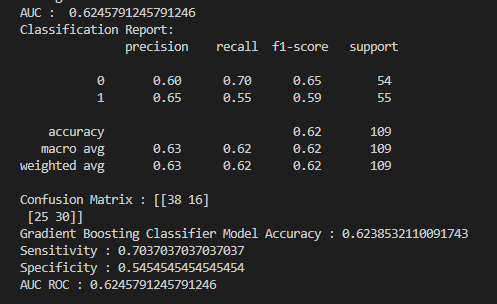
Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Gradient boosting models are becoming popular because of their effectiveness at classifying complex datasets.

İmport all our libraries. I Load in our training data.

Scale our data by creating an instance of the scaler and scaling it:

MinMaxScaler. This estimator scales and translates each feature individually such that it is in the given range on the training set, between zero and one.

****



Accuracy : 0,623853

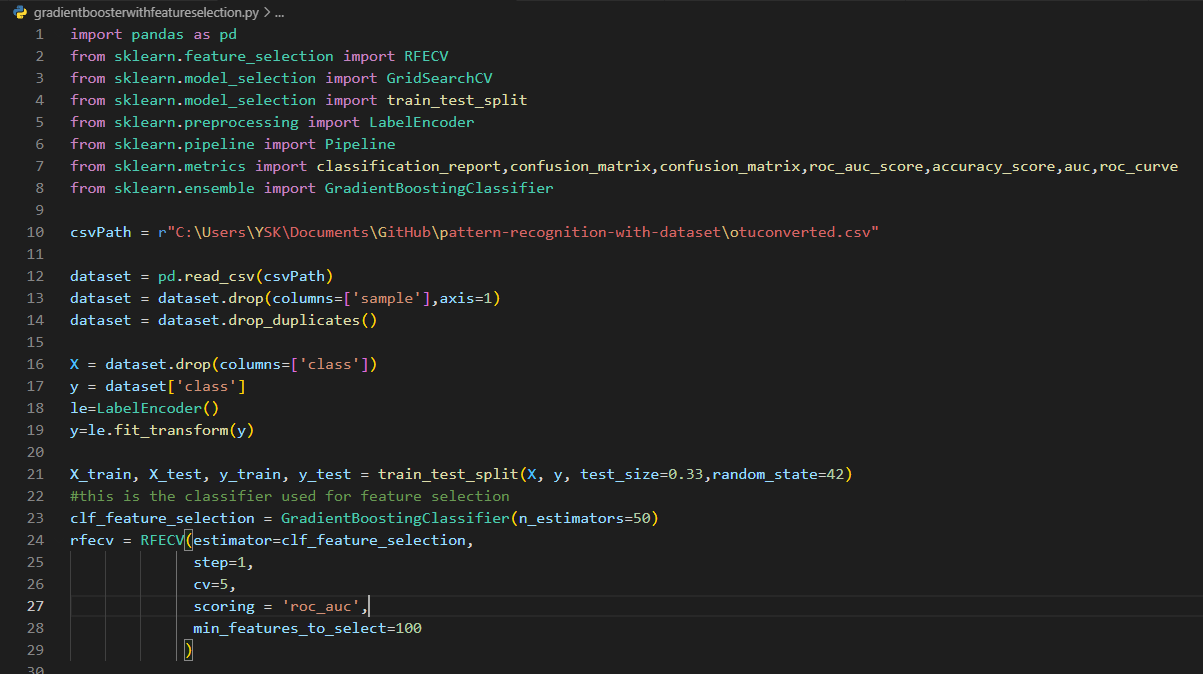
Sensitivity : 0,703703

Specificity :0,5454

Auc: 0,64

**Gradient Booster Algorithm With Feature Selection**

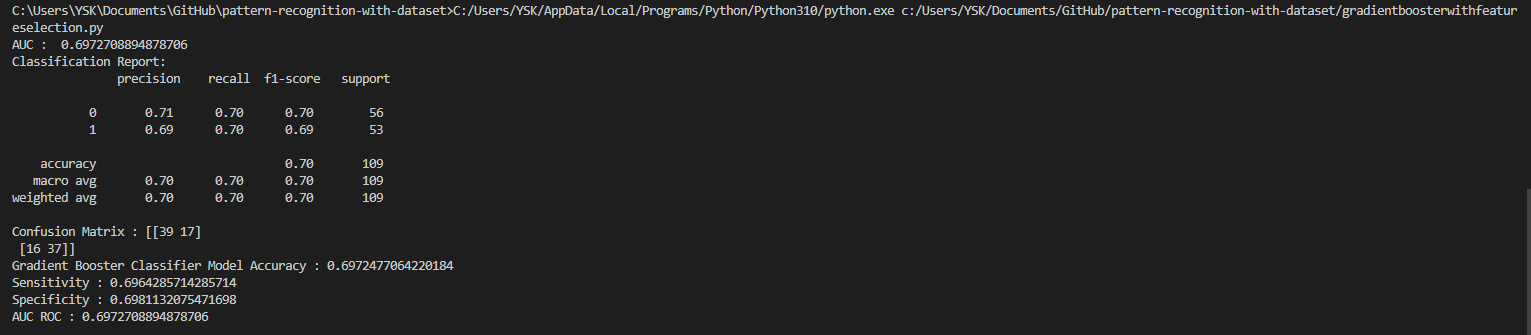
The purpose of the pipeline is to assemble several steps that can be cross-validated and selected features together while setting different parameters.



metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Hyperparemetes are key parts of learning algorithms which effect the performance and accuracy of a model. Learning rate and n\_estimators are two critical hyperparameters for gradient boosting. Learning rate, denoted as α, simply means how fast the model learns. Each tree added modifies the overall model. The magnitude of the modification is controlled by learning rate. The lower the learning rate, the slower the model learns.



Accuracy : 0,6972477

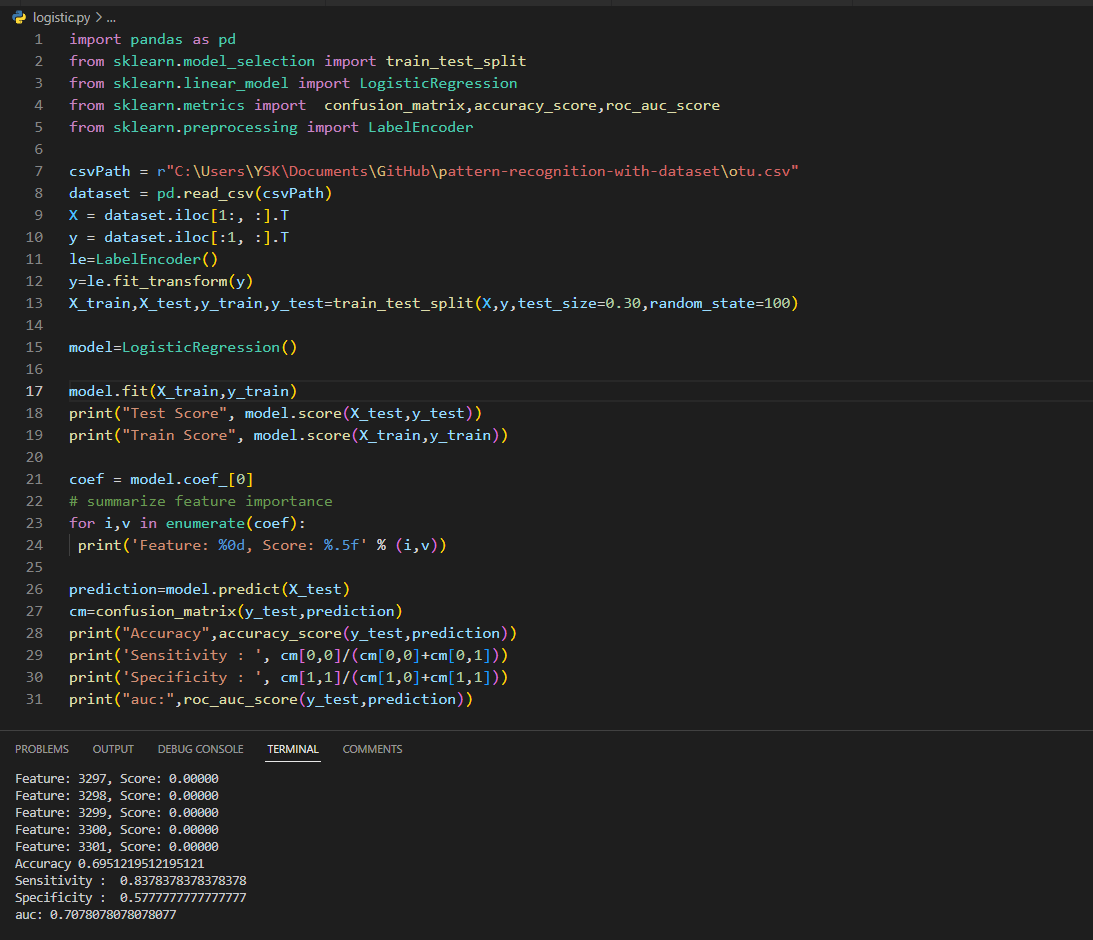
Sensitivity : 0,690

Specificity :0,698

Auc: 0,692

**Logistic Regression Classifier**

Logistic Regression is a classification technique used in machine learning. It uses a logistic function to model the dependent variable.



Test Score: 0.6951219512195121

Train Score: 0.9206349206349206

Accuracy : 0,696

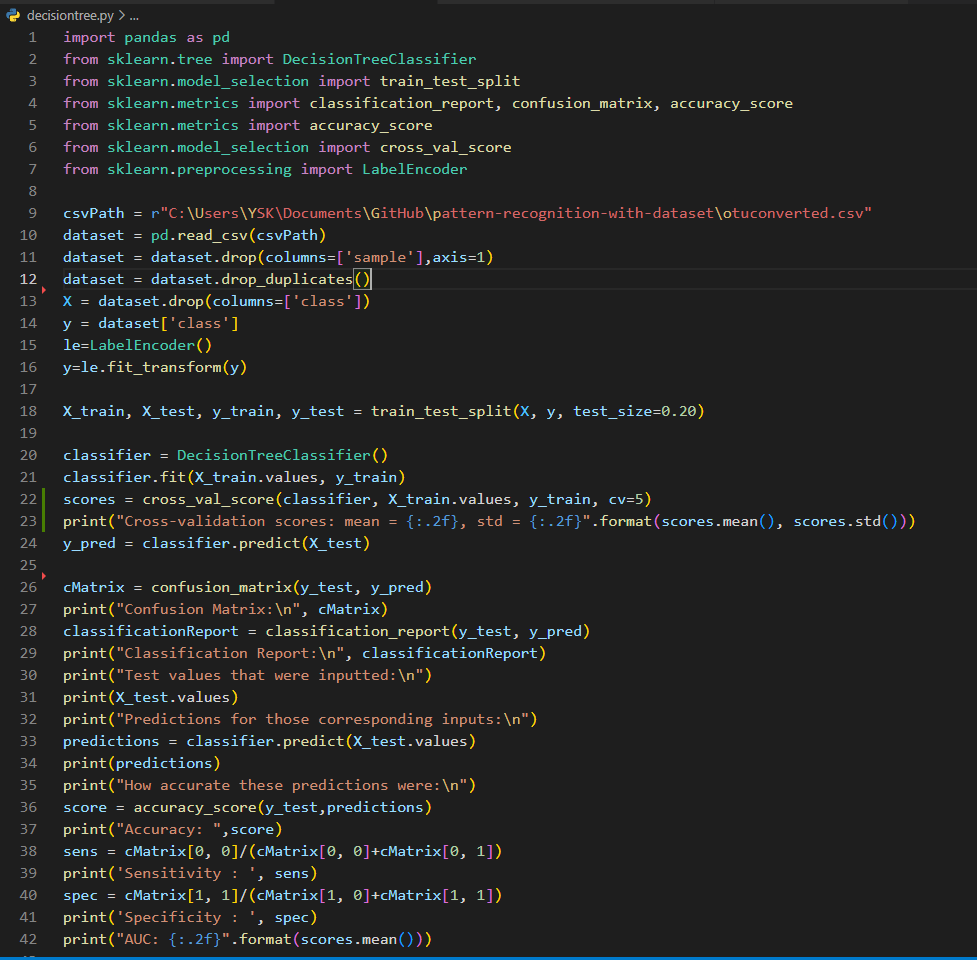
Auc:0,707

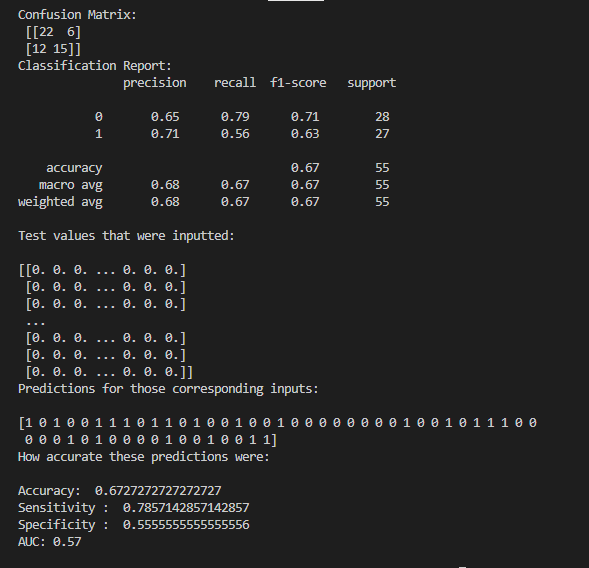
Sensitivity:0,83

Specificity:0,57

**Decision Tree**

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

****

****

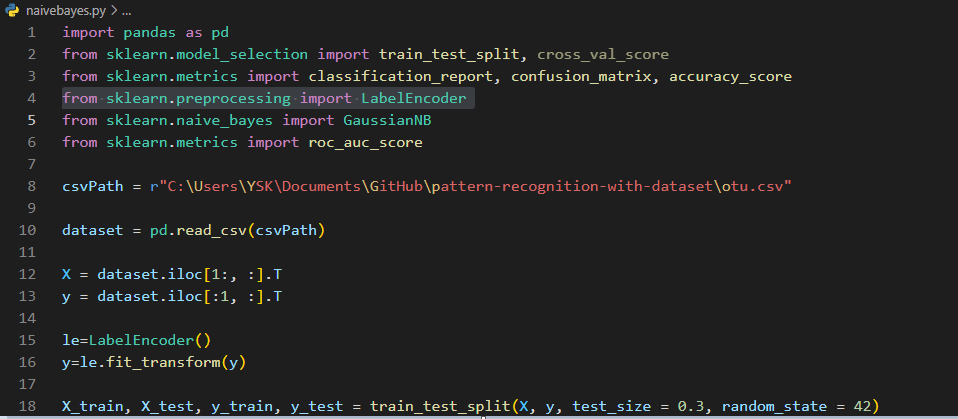
Accuracy : 0,672

Auc:0,57

Sensitivity:0,785

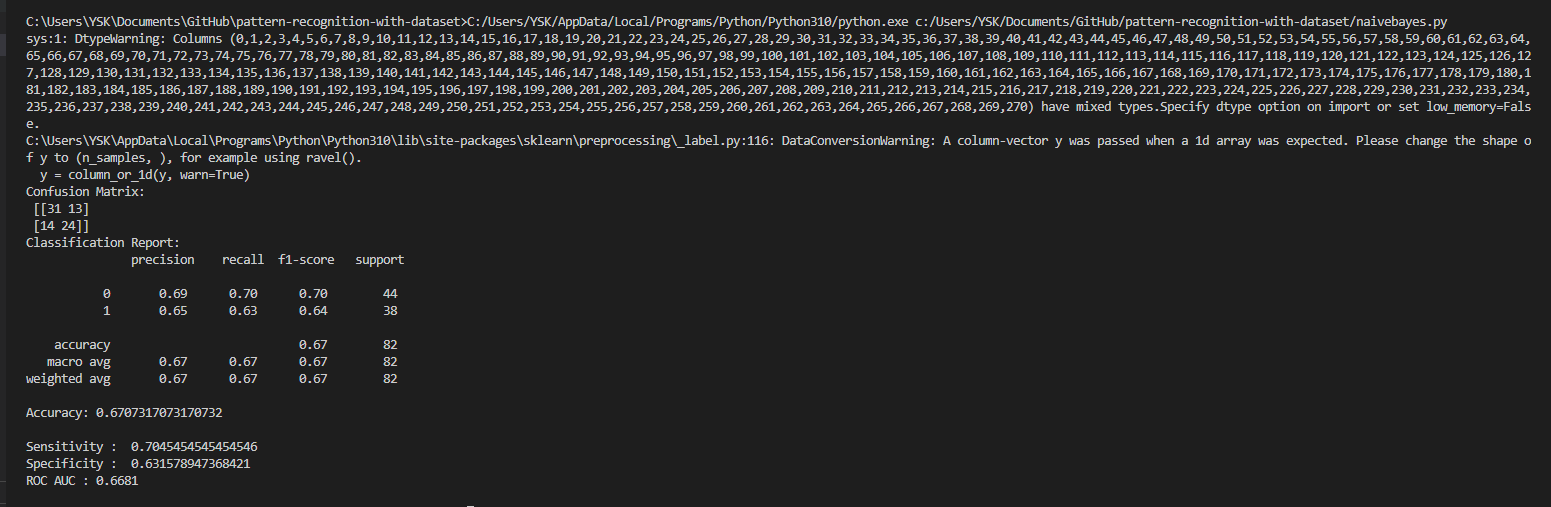
Specificity:0,555

**Naive Bayes Classifier**

****

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

****

Accuracy : 0,670

Auc:0,66

Sensitivity:0,785

Specificity:0,63

**With All the Results**

The random forest algorithm with feature selection gave the best result. The best result does not mean the fastest result.

**Random Forest with Feature Selection Accuracy: 0.733**